APPARATUS AND METHODS FOR INTERCEPTING, EXAMINING AND CONTROLLING CODE, DATA AND FILES AND THEIR TRANSFER IN INSTANT MESSAGING AND PEER-TO-PEER APPLICATIONS

CROSS-REFERENCE TO RELATED APPLICATIONS

[001] This application is a continuation-in-part of co-pending U.S. Serial

No. 09/244,154, entitled "NETWORK TRAFFIC INTERCEPTING METHOD AND

SYSTEM," filed on xx/xx/xx, by Peter V. Radatti and David J. Harding.

FIELD OF THE INVENTION

[002] The present invention relates to apparatus and methods for intercepting,

examining and controlling code, data and files and their transfer. More particularly, the

present invention relates to apparatus and methods for intercepting, examining and

controlling proscribed or predetermined code, data and files and their transfers in instant

messaging and peer-to-peer applications.

BACKGROUND OF THE INVENTION

[003] The rise of the Internet and networking technologies has resulted in the

widespread transfer of code, data and files between computers. This material is not

always what it seems to be. For example, code that is accessed on a remote machine and

downloaded to a computer system can contain hostile algorithms that can potentially

destroy code, crash the system, corrupt code or worse. Some of these hostile algorithms

are viruses, worms, and Trojan horses.

[004] Hostile, malicious and/or proscribed code, data and files ("code" as used

hereinafter generally includes "text," "data" and "files") can infect a single computer

system or entire network and so posit a security risk to the computer system or network.

The user and/or administrator (generally referred to hereinafter as "user") may wish to

1

intercept, examine and/or control such code. The user might also wish to intercept, examine and/or control other code and/or text as well, for example, text which the user does not know to be hostile, but wishes to intercept nonetheless, such as potentially sexually or racially harassing messages, unsolicited messages, etc. This latter type of code is known hereinafter as "predetermined code."

[005] One method of transferring code, instant messaging, has become increasingly popular. Instant messaging may be used for real time text exchanges, as well as other exchanges, such as image transfer, voice and/or video chat, interactive games, code transfers, remote assistance, whiteboarding, code and/or application sharing, etc. Thus a user may transmit code through instant messaging as well as text messages.

[006] The transmission of code through instant messaging may include hostile, malicious, predetermined and/or proscribed code (generally referred to hereinafter as "proscribed code.") Additionally, although messages themselves are usually clear text, encryption may be used on messages and/or code etc., e.g. through Secure Sockets Layer (SSL) and Secure Multi-Purpose Internet Mail Extension (S/MIME).

[007] As with other transfers, it may be desired to secure instant messaging. However, instant messaging applications, such as AOL Instant Messenger (AIM), .NET Messenger (including Windows Messenger, MSN Messenger, etc.), Yahoo Messenger, etc., may be difficult to secure for a number of reasons. For example, in an enterprise installation, IM applications may be installed illegitimately, and thus serve as a conduit for proscribed code. Even those installations that are legitimate may provide, through IM, proscribed code to the user's machine.

[008] Indeed, since Instant Messaging protocols may make use of tunneling and port scanning in order to transmit messages, they may transmit proscribed code without detection. Thus, FTP, Telnet and/or HTTP ports for example may be used by an IM application. As these ports are often left open for other applications, it is difficult to shut them off for IM applications.

[009] Similar problems exist with peer-to-peer and peer-to-peer like applications, e.g., Gnutella (peer-to-peer), Kazaa (peer-to-peer like, as it utilizes a central server.) (Both are referred to herein as "peer-to-peer" or "P2P.") These applications may be installed illegally or without authorization on a system, and may make use of tunneling, port scanning and other techniques making detection difficult.

[0010] Therefore, it would be beneficial to have apparatus, methods and articles of manufacture to simply and effectively intercept, control, and/or examine incoming and outgoing instant messaging and/or peer-to-peer code in an efficient and effective manner transparently or almost transparently to the end-user, with little or no operational effort required by the user.

[0011] It would further be beneficial to have apparatus, methods and articles of manufacture to simply and effectively intercept, control, and/or examine incoming and outgoing secure instant messaging and/or peer-to-peer code in an efficient and effective manner transparently or almost transparently to the end-user, with little or no operational effort required by the user.

SUMMARY OF THE INVENTION

[0012] The present invention comprises apparatus and methods for processing instant messaging and/or peer-to-peer code, that is, for intercepting, examining and/or

3

controlling instant messaging and/or peer-to-peer code in a network. Apparatus embodiments may comprise a protocol parser and a proscribed code scanner. The protocol parser intercepts instant messaging or peer-to-peer code on a communications channel. The protocol parser then transmits the code to the proscribed code scanner for review.

[0013] "Man in the middle" technology is used as well in certain embodiments in order to examine encrypted, e.g., SSL, S/MIME, etc. code. Decryptor/encryptor components permit examination of the encrypted code by a proscribed code scanner once intercepted by a protocol parser.

[0014] Embodiments for various platforms, including Unix® and Windows® NT are disclosed as well. Method embodiments are also disclosed.

BRIEF DESCRIPTION OF THE DRAWINGS

[0015] Figure 1 is a schematic diagram of operation of a preferred embodiment.

[0016] Figure 2 is a schematic diagram of operation of a preferred embodiment.

[0017] Figure 3 is a schematic diagram of operation of a preferred embodiment.

[0018] Figure 4 is a schematic diagram of operation of a preferred embodiment.

[0019] Figure 5 is a schematic diagram of operation of a preferred embodiment.

[0020] Figure 6 is a schematic diagram of operation of a preferred embodiment.

[0021] Figure 7 is a schematic diagram of a prior art embodiment.

[0022] Figure 8 is a schematic diagram of operation of a preferred embodiment.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0023] The present invention comprises apparatus, methods and articles of manufacture for intercepting, examining, and controlling code. The present invention may operate on

a single computer system or multiple systems depending on the operating system and other variables. The preferred embodiments process, that is, intercept, examine, and/or control any or all instant messaging and/or peer-to-peer code transferred through any number of connections in a computer or network. Intercepting, examining and/or controlling instant messaging and/or peer-to-peer code includes but is not limited to monitoring, blocking, logging, quarantining, discarding or transferring code.

[0024] The preferred embodiments monitor instant messaging and/or peer-to-peer code using a protocol parser which may be placed on a client, a server, a peer and/or other systems or components. In the especially preferred Unix® embodiments, the protocol parser is a Unix® STREAMS module and driver activated when an application opens a STREAMS device of the proper type. In the especially preferred Windows® NT embodiments, the protocol parser is a WinNT driver activated when an application opens a communications channel.

[0025] In the especially preferred embodiments, the protocol parser is placed so as to intercept code passing through any channel using TCP and/or UDP (referred to hereinafter as "TCP".) Other possible parameters in the especially preferred embodiments include interception of code traveling to specific instant messaging and/or peer-to-peer port numbers, or code traveling in specific directions (e.g. from an external network to internal network). Code parameters, as well, may be used as interception parameters.

[0026] The intercepted code is sent by the parser to a protocol scanner. The protocol scanner in turn reviews the code it has received from the parser. Depending upon its configuration, which in the preferred embodiments include port interception designation,

5

file type interception designation, instant messaging content and/or other parameters, the protocol scanner will pass some, all, or none of the code flowing through it to a proscribed code scanner. This proscribed code scanner may be an antivirus scanner, pattern scanner, and/or content scanner or other types. This proscribed code scanner, depending on its settings, may pass some, all or none of the code passing through it.

[0027] If the code passing through the intercept module is not intercepted, that is, passed to the protocol scanner, it is returned to the communications channel. Similarly, if the code passing through the protocol scanner is not intercepted, that is, passed to the proscribed code scanner, it is returned to the communications channel.

[0028] Thus, the embodiments of the present invention process code by intercepting or diverting code from a communication channel, processing it, and then reintroducing it (or not as desired) into that channel. By intercepting or diverting code from a communication channel, the preferred embodiments do not change the nature of the communication, and process code transparently to the applications, unless a problem is discovered. In that case, the preferred embodiments may, if desired, be configured to notify the application and/or the user, among other actions.

[0029] When the parser intercepts a request from a client or server for an SSL transfer the parser creates a new SSL server that communicates with the original client and a new SSL client that communicates with the original server. The SSL server and SSL client may then intercept any and all communications that occur between the original SSL client and original SSL server.

[0030] The preferred embodiments will also intercept and review S/MIME messages. The S/MIME messages will be intercepted by the protocol parser and sent to a S/MIME

6

decryptor. The decrypted messages can then be scanned by a proscribed code scanner, which may be an antivirus scanner, pattern scanner, and/or content scanner or other types. The proscribed code scanner will then review the code and signal whether the S/MIME message may be released from interception.

[0031] A preferred embodiment of the present invention runs on a Unix® platform that supports STREAMS such as System V, Sun Solaris®, IBM AIX®, HP-UX®, etc. The use of STREAMS in this embodiment promotes ease of installation and use. For example, installation on a computer of this STREAMS embodiment means the Unix® kernel on the computer would not have to be recompiled as might be the case with a non-STREAMS embodiment.

[0032] The following description of the embodiments uses Sun Solaris® operating system Unix® terminology. However, it should be specifically understood that embodiments can be implemented in other Unix® and Unix®-like platforms, including but not limited to Linux® and its variants, OS X, as well as other operating system platforms and code including but not limited to Microsoft Windows® XP, Windows® NT, Windows® 2000, Windows® 95, 98 CE and Me, IBM OS/390, MacOS, VxWorks,® Palm OS, Symbian, Java and others. Although the present invention can be implemented on various platforms, the preferred embodiments are used in Unix® and various Windows® environments, such as NT, 2000, 95, 98 and Me.

[0033] Moreover, those skilled in the art will appreciate that the invention may be practiced with other electronic device and computer system configurations, using TCP/IP or similar communications protocols, such as wired and wireless handheld devices and/or computers, cell phones and other wired and wireless communication devices, digital

audio and video devices, distributed computing environments, multi-processor systems, and microprocessor-based or programmable consumer electronics, such as smart printers, network PCs, minicomputers, mainframe computers, and the like. For example, embodiments may be used to process code transmitted via short messaging services or other cellular services.

[0034] The preferred embodiments are comprised of both platform dependent and platform independent code as well, thus increasing portability and simplifying porting to new platforms. Generally, the embodiments will comprise both types of code although it is possible and within the scope of the present invention to construct an embodiment wholly in platform dependent code. Similarly it is possible and within the scope of the present invention to construct an embodiment wholly in platform independent code.

[0035] The preferred embodiments that run in a Unix®-STREAMS environment are generally invisible. At least part of those embodiments operate within the Unix® kernel, and so a brief review of the operation of the kernel would be helpful to understanding those embodiments.

[0036] The Unix® kernel is the operating system. The kernel is invisible to the user yet controls the system resources and hardware. For example, the user may want to use a telecommunications program in order to call another computer. She opens the program and instructs the program to call a mainframe. The program in turn issues a call to the kernel to open the communications device, in this case a serial port device. The kernel receives the call and looks up the device in its character device switch table. The character device switch table provides the kernel with the appropriate device specific

8

routine: in this case an open call. Subsequent commands from the program will be similarly interpreted by the kernel.

[0037] In a Unix®-STREAMS platform, any specific devices with a STREAMS implementation (a STREAMS device) have a special field in the character device switch table that points to a streams initialization structure. Opening the STREAMS device will open the STREAMS device driver as well as create a Stream head to access the device driver. One or more STREAMS modules may be inserted between the STREAMS device and Stream head. STREAMS modules can process code passing through the Stream. If, for example, taking the above example regarding the opening of a telecommunications program, the user opens her program in a Unix®-STREAMS environment with a STREAMS serial port device, when the program calls the kernel to open the serial port device, a STREAMS serial port device driver will be opened and a Stream head will be created. A STREAMS module may be inserted as well depending on the configuration of that particular Stream. STREAMS modules are generally inserted into the Stream automatically through use of an "autopush" list which is referred to by the kernel when a Stream is opened.

[0038] Returning now to the description of the preferred embodiments, installation of this embodiment begins with installing tpicept, a kernel module, in an appropriate directory or directories, in order to provide the module to the kernel when called. In the especially preferred embodiments, tpicept will provide both a STREAMS module as well as a STREAMS device driver to the system as is discussed in further detail below. tpicept is linked to a number of path names, so that the file can be called by a number of names and used in a number of ways, e.g., links are established in /usr/kernel/drv and

9

/usr/kernel/strmod so that the file may be used both as a STREAMS driver and STREAMS module. The protocol scanner cyb_protocold and the proscribed code scanner are also installed at this time in appropriate directories, e.g., /usr/sbin.

[0039] When installing the preferred embodiments, the "old" or existing autopush lists for the TCP protocol layer and a new list containing the STREAMS module tpicept is added. Thus tpicept will be pushed onto the stream that is created when a TCP streams device is opened.

[0040] The embodiments may be started by any of a number of manners as is known in the art. For example, no separate procedure is necessary to start the operation of the kernel module tpicept. It operates as soon as called.

[0041] A protocol scanner and proscribed code scanner, as described further below, may also be made operational through an executable file. The configuration of the preferred embodiments also allows for loading on startup of the system. Of course, other embodiments may load when desired by the user. However, loading upon start-up limits user tampering with the security procedures which may be desirable for network administration.

[0042] Once installed, the embodiment is operational. The embodiment remains in the system until a STREAMS device – such as an instant messaging or peer-to-peer process – that the embodiment has been configured to intercept is opened. The especially preferred embodiments intercept code at the TCP layer, however, the Unix® embodiments of the present invention are not limited to interception at the TCP layer. Rather, the Unix® embodiments of the present invention can intercept, examine and/or control code in any

10

Stream. Of course, other embodiments, on other platforms, may also intercept code at other locations on a communications channel or layers in a communications stack.

[0043] For example, in preferred Unix® embodiments, when a Web browser or other application opens a STREAMS device, a STREAMS module tpicept will be inserted in the stream. Figure 1 shows an example of a process of a preferred embodiment. As shown in Figure 1, tpicept intercepts code passing in the stream. (The remainder of the stream or communications stack is not shown here.) (If the Web browser opens a secure connection, for example the HTTPS connection shown in the figure, the tpicept module will detect the opening and call a protocol SSL server and protocol SSL client, as is described in more detail below.)

[0044] Code passing back and forth between the Web browser and the NIC STREAMS device is being intercepted at the HTTP-TCP interface by the STREAMS module tpicept. tpicept, may or may not transfer the code to a protocol scanner, *cyb_protocold*, depending upon, in this embodiment, specific configurations that include port interception settings. (The dashed line in the figure shows this optional transfer.) For example, if *tpicept* has been configured so as to only intercept AIM (America OnLine Instant Messaging) communications on port 5190 (as a default), only code passing through port 5190 will be sent to *cyb_protocold*. Code traveling to another port (e.g., through SMTP on port 25 from a mail application) will be returned to the communication stack.

[0045] It should be noted that the preferred embodiments scan incoming code packets by examining the header information contained on those packets. For example, in TCP/IP protocol the internal address of any particular packet within a system (including port

11

number) is contained in the header. By reviewing the header, the protocol scanner of the preferred embodiments can intercept packets sent to a predetermined port.

[0046] The STREAMS module may also be configured by varying its insertion at any particular data communications layer and/or protocol. In this embodiment, the module is inserted to intercept code at the upstream side of TCP. The STREAMS module has been inserted here in this embodiment because TCP is a multiplexer for different applications and connections, including those applications and connections to the Internet. Thus, this embodiment intercepts traffic at a "choke point" of the system. Of course, in other embodiments, other protocols and/or other layers may be intercepted as desired.

[0047] Returning to Figure 1, if *tpicept* intercepts the code according to its configuration, it sends the code (through a STREAMS queue pair not shown here) to the protocol scanner, *cyb_protocold. Cyb_protocold* then may transfer the code to the proscribed code scanner through interprocess communication, e.g., a pipe. (The dashed line in the figure shows this optional transfer.) The decision by *cyb_protocold* to transfer the code depends upon its configuration settings and these settings can be varied as desired in this embodiment.

[0048] For example, a default setting for AIM (American Online Instant Messaging) may be port 5190. Thus, the scanner may be configured to intercept code on specific TCP ports such as port 5190. (As should be clear, both the protocol scanner and the STREAMS module can be configured by port interception settings). Moreover, the scanner may also be configured to intercept certain file types attached to the instant message, such as *.exe files, etc.

12

[0049] As another example, the protocol scanner could be configured as a protocol dependant scanner, that is, to presume that traffic on particular port will always be a certain protocol. Those ports which have not been configured for interception by this embodiment will pass traffic transparently. Alternatively, "protocol independent" scanning, that is, scanning by ignoring the actual port used, could be configured. Protocol independent scanning provides a higher level of security and may be desirable in some cases when non-standard ports are used for well-known protocols (such as changing the AIM port from its default port 5190.)

[0050] Protocol independent scanning embodiments may be especially useful in intercepting, controlling, and/or examining incoming and outgoing instant messaging and/or peer-to-peer code because of the ability of these programs to use other than their well known ports. That is, instant messaging and/or peer-to-peer programs can switch ports as they are being used. Thus, embodiments using direct examination of the code will be able to detect code transfers to and from instant messaging and/or peer-to-peer programs.

[0051] Direct data examination may occur in number of ways. For example, direct data examination may occur through examining the initial handshake of code sent to and/or from the instant messaging or peer-to-peer program; examining header(s) in the code, examining code format, etc. In the preferred embodiments using direct data examination, a code decryptor such as that presently available from CyberSoft, Inc. under the name UAD® may be used to deconstruct the code for review by the proscribed code scanner.

[0052] It should be noted that configuration may be manual or automatic, so that, for example, upon installation of more than one instant messaging applications, the

13

configuration is set to include those ports and other parameters that differ between the two applications.

[0053] If the code is not sent by the protocol scanner to the proscribed code scanner, it is returned to the STREAMS module, and then to the Stream.

[0054] If the code is sent by the protocol scanner to the proscribed code scanner, (e.g., a proscribed code scanner such as that presently available from CyberSoft, Inc. under the name VFIND®) the proscribed code scanner available the proscribed code scanner analyzes the code according to its configuration settings. These settings can take numerous forms, including scanning for proscribed code or code segments, calculating hash codes of the code or code segment, etc.

[0055] If the proscribed code scanner scans code segments or calculates hash codes, it compares the result(s) to one or more database(s) of proscribed code samples. If the result(s) match the samples, and therefore the code is not acceptable, then a number of options may be available, as is described further below.

[0056] For example, the user might wish to intercept, examine and/or control sexually harassing text. Any instant messages might contain text that might be offensive, and so the user can provide words and/or terms (i.e. code) that she might wish to be identified as sexually harassing to the proscribed code scanner. Moreover, embodiments may be utilized for control of messages received by another, such as when a parent wishes to scan messages received by her child. Thus, configurations are supplied enabling for the scanning of preconfigured words as well as user supplied words.

[0057] The proscribed code scanner is provided with one or more proscribed code database(s), which may be modified by the user in the preferred embodiments. Thus,

14

flexibility is provided to the user. (The databases are secured against unauthorized changes, in manners known in the art.) Moreover, in certain embodiments, preconfigured databases provide areas of proscribed code, e.g. a database for sexually harassing code, racially harassing code, age appropriate code (e.g. 3-8, 9-12, etc.), etc.

[0058] It may be desired, as well, to provide for manual and/or automatic downloading of updated and/or optional databases as well. In this way a user can tailor the product to her use.

[0059] If the code is acceptable, the proscribed code scanner provide an indicator to *cyb_protocold* which in turn will send the code back through *tpicept* and from there back to the communication stream. In this case, the process will have been transparent, that is, the interception process will not have disturbed the application. Thus, this embodiment has minimal impact on the computer and networking connections.

[0060] If the code is not acceptable, the proscribed code scanner will indicate the code is not acceptable. In the especially preferred embodiments, that indication will be sent directly to the user and/or the application. In other embodiments, the indicator may result in access to the code being denied; in extracting the proscribed code from the remainder; in quarantining and saving or transferring the proscribed code for analysis or deleted; the proscribed code could be modified; the proscribed code could be sent to an administrator or security department or firm; etc. The code may be sent to another communications channel, whether or not acceptable.

[0061] Code may be also certified by a certification component. Thus, if the code is acceptable, it may be sent from a scanner accompanied by a certification mark or other

designation (e.g. message stamp, authentication, registration, etc.) Notification of certification may also be sent to the sender as well.

[0062] Embodiments may also translate intercepted code. For example, if code sent to an unauthorized instant messaging program on a system is intercepted, and approved for transmittal, yet, for security reasons the instant messaging program is not permitted, and necessary program parameters may be translated (e.g., protocols, formatting, etc.) so the code may be received by an authorized program, such as email, authorized instant messaging programs, etc.

[0063] The process of this embodiment is shown generally at Figure 2.

[0064] It should be noted that more than one instance of the embodiment will be utilized if the user has more than one application running and/or more than one communications channel opened. For example, if a user opens an instant messaging application over a network card communication channel as well as a sendmail application over a network card communication channel two code streams will be created, both using TCP. Both will be intercepted according to the process of the preferred embodiments. Moreover, if channels are opened using other protocols at the Transport layer, or other protocols at other layers, embodiments of the present invention can be used to intercept those communications, with an appropriate kernel module.

[0065] Of course, any configuration parameters are not limited to predetermined parameters. For example, the interception parameters of the intercept module, protocol scanner and the proscribed code scanner may be configured in various ways in various embodiments. Moreover, the user may configure and reconfigure the parameters as desired. In yet other embodiments, there may need to be no interception parameters –

interception can be turned off – or no predetermination of interception parameters. For example, a user may decide to intercept all code, or the embodiment could request parameters as code is being transferred or could request an interception decision as the code is being transferred. Of course, in yet other embodiments, the choice of predetermined, non-predetermined, or no interception parameters, and what parameters to change could be offered to either or both end-users or network administrators.

[0066] Of course, other embodiments may be configured differently. For example the protocol scanner may be written as one or more STREAMS modules, and the connections to and from the intercept module and the proscribed code scanner would be modified appropriately.

[0067] Another especially preferred embodiment is written for the Windows® NT platform. In this NT embodiment, installation of this embodiment begins with installing csiservice.exe in an appropriate directory or directories, e.g., WinNT\System32\, as well as csitcpip.sys, an NT driver, in an appropriate directory or directories (usually, WinNT\System32\drivers). csiservice.exe provides the driver to the system when called. The protocol scanner and the proscribed code scanner are also installed at this time in appropriate directories, e.g., WinNT\System32\.

[0068] After rebooting, csiservice.exe calls the driver, which in turn, will call the protocol scanner if code is transmitted through a communications stack which the driver has been configured to intercept. The interception settings and processes, which may be set in any of a number of ways, are as described above with regard to the UNIX® preferred embodiments. Figure 3 shows an example of a process of a preferred WinNT embodiment intercepting code in a NIC communications stack.

[0069] As had been described above, embodiments may intercept encrypted code. Embodiments may also encrypt code after interception. For example, an entire communications stream may be encrypted by using a VPN-protected session (such as a PPTP, L2TP, or IPSec tunnel.) A message and/or other code, e.g. a file, might be encrypted as well. Therefore, embodiments are provided to process altered code. Altered code is defined herein as code altered by encryption or a communication protocol after being generated by an application or program for transmission to a complementary encryption or communication protocol.

[0070] In order to intercept the altered code, embodiments are used in a "man in the middle" implementation. For example, both SSL and S/MIME alter code by securing code through encrypting and decrypting code on both the server and client ends of the communication with a fixed key. SSL is a protocol layer encryption used in TCP connections and implemented between the HTTP layer and the TCP layer. Code is encrypted as it passes through the SSL protocol layer. S/MIME secures code by encrypting mail messages and their attachments through a mail application.

[0071] In the preferred Unix® embodiments, when an instant messaging or other application opens a STREAMS device, a STREAMS module tpicept will be inserted in the stream. As shown in Figure 4, tpicept intercepts code passing in the stream. (The remainder of the stream or communications stack is not shown here.) If the instant messaging application opens a secure connection, for example the HTTPS connection shown in the figure, the tpicept module will detect the opening and call a protocol SSL server and protocol SSL client, as is described in more detail below.

18

[0072] The tpicept STREAMS module begins the interception when tpicept detects a connection to a specific port, such as a port commonly used for instant messaging. tpicept opens a channel to a protocol SSL server instance in order to send code to the protocol SSL server. tpicept will also open a protocol SSL client instance. The protocol SSL server and the protocol SSL client will continue with the usual SSL negotiations, including the choice of encryption keys, thus placing themselves in the position of the SSL server that the user client application expects to communicate with and the SSL client that the SSL network expects to communicate with.

[0073] Turning to Figure 5, once the protocol SSL server and protocol SSL client are called, all communications through the stream will pass through those connections. (The remainder of the stream or communications stack is not shown here.) The protocol SSL server will decrypt the communications and, in the especially preferred embodiments, send the communications to a proscribed code scanner. The proscribed code scanner will in turn scan the code, as was described in the above embodiments with regard to the proscribed code scanner of Figures 1-3.

[0074] In the preferred embodiments, implementing a single SSL connection as shown in Figure 6, the user client application identifies a single remote site with a single server certificate, such as shown in the prior art schematic diagram of Figure 7. Thus there are two SSL connections opened on either side of the "man in the middle," as can be seen in the schematic diagram of Figure 6, rather than one, as would be the case without the "man in the middle" interception, as can be seen in the prior art schematic diagram of Figure 7.

[0075] The user's certificate database – necessary for a SSL connection – usually is managed through the user client application. In some embodiments, this database cannot be modified directly. (However, it may be possible in other embodiments to modify this database directly.) In some embodiments, automatic verification of site specific certificates may occur, however, certificates from certificate authorities may not be accepted. If a certificate from an untrusted site is received, an notification component of the embodiment may notify the user and/or administrator of the untrusted certificate and requesting confirmation. If the user and/or administrator indicates that the certificate should be stored, then the notification component will store the certificate in its own global certificate database. Alternatively, the user or administrator could then import this certificate into the instant messaging database. If a certificate authority provides an unknown certificate, the certificate could be added to the instant messaging database through manners known in the art.

[0076] Figure 8 shows the preferred embodiment in the course of intercepting S/MIME code transfers. An S/MIME instant messaging application has been opened and the tpicept STREAMS module intercepts all code sent to and from the application by inserting itself on the communications stack as described above with regard to SSL interception. The tpicept module has called an S/MIME encryptor component and an S/MIME decryptor component, which in some embodiments are installed along with SSL server/client components similar to those described above. S/MIME code is then sent to the S/MIME decryptor component by the intercept module which decrypts the code in order to then send the code to a queue, where the code awaits scanning by a proscribed code scanner.

[0077] Proscribed code scanning in any of the altered code embodiments proceeds in a similar fashion as was described above with regard to the proscribed code scanner of Figures 1-3. If the code is acceptable, the proscribed code scanner will indicate to the protocol SSL client, S/MIME encryptor, other encryptor, etc. that the code is acceptable and can pass through the system. The protocol SSL client, S/MIME encryptor, other encryptor, etc. that, will re-encrypt the code, and return the code to the communications stack through the tpicept module. The process will have been virtually transparent, that is, the interception process will have minimal impact on the performance of the computer and networking connections.

[0078] If the code is not acceptable, actions such as were described in the above embodiments with regard to the proscribed code scanner of Figures 1-3 will be taken.

[0079] The proscribed code scanner analyzes the code according to its configuration settings. These settings take numerous forms, similar to those described above with regard to the proscribed code scanner of Figures 1-3. Similarly, configuration is similar to that described above.

[0080] It should be noted, in order for the S/MIME decryptor component to decrypt the mail, the private key of the sender or recipient of the mail must be located and used. If the key is not found, the mail is stored or quarantined in this embodiment and the user or administrator will be notified to supply the key or its location. When the embodiment stores or quarantines received mail in this embodiment, the sender or originator will have no direct indication that this has occurred. Therefore, the storage or quarantine area is preferably checked manually. Other embodiments may notify the user, through methods known in the art, that intercepted mail is waiting in the received storage area.

21

[0081] Other preferred embodiments are written for the Windows® NT platform. In the especially preferred NT embodiment, installation of this embodiment begins with installing csiservice.exe in an appropriate directory or directories, e.g., WinNT\System32, as well as csitcpip.sys, an NT driver, in an appropriate directory or directories (usually, WinNT\System32\drivers). csiservice.exe provides the driver to the system when called. The protocol scanner and the proscribed code scanner are also installed at this time in appropriate directories, e.g., WinNT\System32.

[0082] After rebooting, csiservice.exe calls the driver, which in turn, will call the protocol scanner when a Web browser or other application opens a secure connection. The interception settings and processes, which may be set in any of a number of ways, are as described above with regard to the UNIX® preferred embodiments.

[0083] The man in the middle embodiments may be used in any of a number of interception environments, and those embodiments may be constructed in accordance with the present invention, including interception of encryption and other communication protocols known and unknown in the art. Certain embodiments may as well combine interception processes, e.g. combining both SSL and S/MIME interception processes in one embodiment, or other combinations of encryption or other communication protocols. It is also possible to encrypt messages that might have been originally encrypted, as well as the reverse.

[0084] It should also be noted that the man in the middle embodiments are not limited to use of a protocol parser, such as a Unix® intercept module, WinNT driver or other interception means. For example, embodiments implemented on a local instant messaging application may replace a local delivery agent with S/MIME

decryptor/encryptor components which in turn will have the code scanned by a proscribed code scanner as described above.

[0085] The preferred embodiments may be used on a single machine, with a connection to another machine or system, network or the Internet. Preferred embodiments may also be used on a separate machine or system inserted in a communications channel, including but not limited to another computer, proxy server, firewall, router and the like. Embodiments may also be invisible to the user, be operational in stealth mode, etc. Results may be logged, automatically and/or manually.

[0086] The above description and the views and material depicted by the figures are for purposes of illustration only and are not intended to be, and should not be construed as, limitations on the invention.

[0087] Moreover, certain modifications or alternatives may suggest themselves to those skilled in the art upon reading of this specification, all of which are intended to be within the spirit and scope of the present invention as defined in the attached claims.